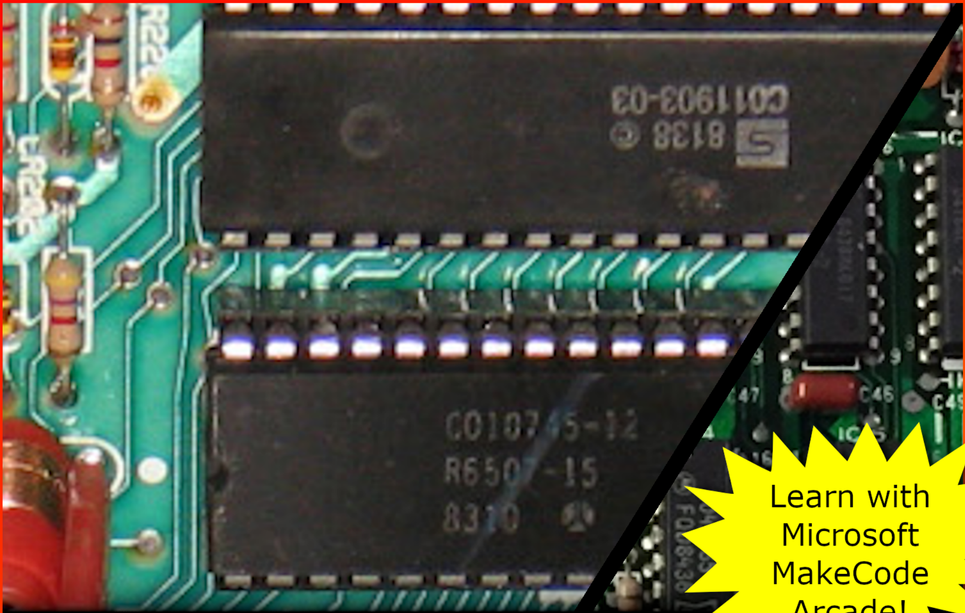
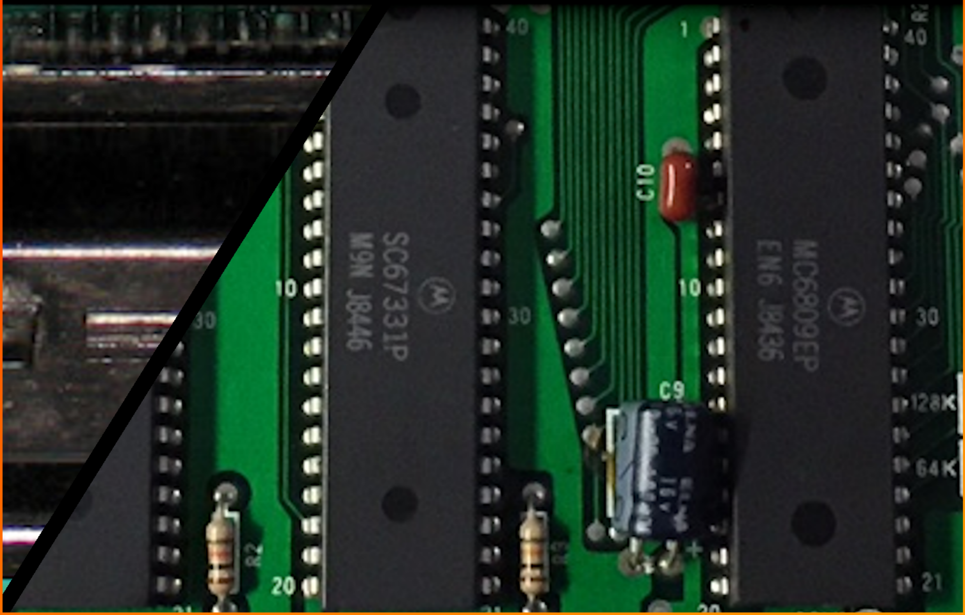


Learn with Microsoft MakeCode Arcade!

Robo Teaches Programming
Part Zero: A History of Video Games
Part Zero Mark II: Time to Type!

Alex Kulcsar



Robo Teaches Programming Part Zero Preview

A prelude using Microsoft MakeCode Arcade

Alex Kulcsar

This book is for sale at http://leanpub.com/makecode-book

This version was published on 2022-08-04

Contents

Notes about this preview edition

Update: August 2022

Hi, all! I bet you thought I forgot about this project. Far from it! I really miss working on this book. I have, though, been writing … about MakeCode Arcade, to boot!

You can learn about a course that I wrote with another author at the MakeCode Arcade home page. Scroll down to the **Courses** section, and you'll find a new course named *Introduction to Computer Science with MakeCode Arcade*. The curriculum is available for free on GitHub to verified educators.

The pandemic interrupted my writing schedule, but I'm returning to my personal projects now that we have wrapped up the Microsoft project. This update includes the next chapter on variables and data types. I've also added the raw text for a couple of new appendices taken from posts that I've made in the MakeCode forums. The outline also reflects two new chapters on debugging.

I recently started using the LaTeX typesetting system, and I have fallen in love with it. I will be taking another pause from adding new content to this book while I convert the existing text to LaTeX. As I do so, I'll be writing about my journey and the tools that I use in another book. If you're interested, you can follow my progress here:

https://leanpub.com/latex-tech-pub

That book will be available free of charge on Leanpub. The next preview of this book will be typeset with LaTeX, and the online course should be ready for you to preview, too.

Stay tuned!

This preview edition of *Robo Teaches Programming* will differ from the final version. Some things to note:

- I am using Leanpub to render the preview editions. Final versions will be translated into Bookdown and will use a Tufte-style layout. I also am considering transitioning to MarkDeep to save the translation steps later.
- Leanpub does not like nested ordered (i.e., numbered) lists. Hopefully, Bookdown handles those better. :-) MarkDeep certainly does, and I may switch to that, even for rendering the preview editions.
- Some images are merely placeholders and will be replaced with appropriately-licensed images. Those images are noted in the text.
- Figures, tables, and exercises are mentioned with the generic "Figure/Table/Exercise A-B," and the actual labels are not printed in the captions. Leanpub does not automate the numbering of figures, tables, and exercises. That will be handled during prepress or in my transition to MarkDeep.
- In-text citations and their related references are formatted neither correctly nor consistently. That also will be handled during prepress.
- Bibliography is in BibTeX syntax. Appropriate formatting will be handled during prepress.
- Link code site has not yet been built. A list of link codes and their URLs is provided at the end of the book for the time being.

As noted at the bottom of most pages, this preview copy is provided for your own, personal use. Do not distribute in whole or in part without express, written consent. © 2019-2023 Robo Technical Group, LLC. All rights reserved.

Preview gifts

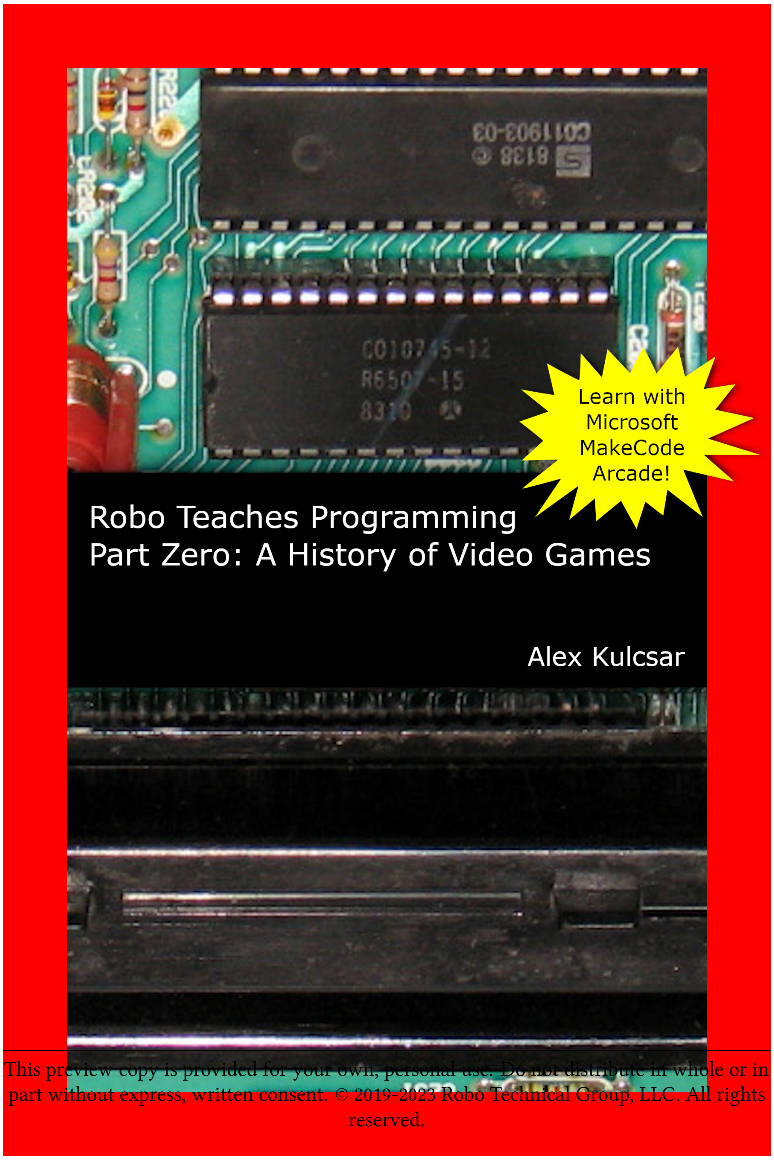
As members of the preview community, in addition to continued access to this e-book, you will receive several gifts to reward your early support of this project:

- Access to exclusive versions of the online courses based on these books.
  - Enrollment code: **LeanPub-Thanks!-2021**

- Autographed copies of the first printings of these books.

I will be managing fulfillment of these gifts through an LMS that will house online courses for this series. Keep an eye on this space for updates on your preview goodies!

Part zero: A history of video games


Learn with Microsoft MakeCode Arcade!
Robo Teaches Programming
Part Zero: A History of Video Games
Alex Kulcsar

Cover image for "the red book"

About the cover

As with many families in the United States during the 1970s, my family's first computer was the Atari Video Computer System (also known as the Atari VCS or, most commonly, the Atari 2600). My siblings and I spent many hours playing video games on that Atari 2600, usually connected to a black-and-white, nine-inch television in one of the bedrooms. Every once in a while, our parents would let us connect it to the large, color television in the living room.

The cover image is a photograph of a portion of the circuit board of the Atari 2600 that I currently own. In the photograph, you can see the MOS Technology 6507 CPU.

Screen shot of Moon Patrol on the Atari 2600 (Temporary image from Atari Age)

ℹ Cover images are temporary and need to be replaced.

Brief contents

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

A history of video games

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Introduction to *Part zero*

About the series

Robo teaches programming is a six-part double-trilogy! We start slowly, introducing you to basic programming concepts with a focus on graphical applications. In the later parts of the series, we introduce you to some more advanced concepts that are fundamental to computer science.

Intended audience

The series is meant to be used in undergraduate college courses, although the books can be used in a high school setting or by anyone else with an interest in programming. Part zero is meant to be used in a CS0-style course; parts one and two for CS1; parts three and four for CS2.

Titles in the series

- *Part zero: A history of video games* ("The red book." That's this book!)
- *Part zero mark II: Time to type!* ("The orange book.")
- *Part one: The basics* ("The yellow book.")
- *Part two: Beyond the basics and introduction to OOP* ("The green book.")
- *Part three: Advanced OOP* ("The blue book.")
- *Part four: Introduction to data structures and algorithms* ("The purple book.")

About the book

The full name of this book is *Robo teaches programming part zero: A history of video games*. That is a lot to say and a lot to type! So, for simplicity, I affectionately call this book "the red book."

In the red book, we start gently and get our feet wet early in a great environment: Microsoft MakeCode Arcade. We begin by exploring an existing game within the environment. By the end of the first chapter, you will have created your first masterpiece! We use the Blocks interface throughout this prelude, yet we learn to create some pretty sophisticated programs. Throughout the book, we also stroll through the history of video games to give our projects some additional context.

History of video games

We study and create a lot of video games in this book. We use the history of video games in this book to give some context to the skills that we build. In each chapter, we look at the story behind a video game or platform that is relevant to the skills presented. Those stories will be presented in "aside" boxes like this one.

Why video games?

When I was learning to write computer programs, some of the first programs that I wrote were games. Even now, when I am learning a new programming language, I often use simple games to help me learn the language. I also use video games and board games as diversions. I play video games on a console, on a PC, and on mobile devices. I play board games, party games, and video games with friends and family. I also create games as a diversion. Writing these games helps to keep my programming skills sharp, and often I use them as a way to learn new skills. With creating games, I've learned the more obvious skills, like adding graphics to my programs and moving them around on the screen. I've also learned how to use databases and other data files when writing games, along with other skills that you may not associate with games.

Since I started writing this book, I've written a *lot* more games! You'll get to see some of them as we travel on this journey together.

As a computer programmer, I am not alone in my love of games. But, *why* are many computer programmers fascinated with games? I've heard several explanations for this, but this quote from Hennessey and McGowan's *The comic book story of video games* resonates the best with me, where they discuss that computer games have been around since the very beginning.

The first slate of video games all came about as explorations *into*

> the capabilities of this unproved frontier.... One such game was *draughts* ("checkers" in North America), realized in 1951 in Manchester, England. The computer that ran *draughts*–the von Neumann architecture-based Ferranti Mark I had its programs coded on and loaded in via hole-punched paper tape.... The very existence of these games sort of goes against the grain of what computing was first supposed to be about.... Yet many programmers kept returning to making–and playing–*games*. Often anyone daring this felt themselves placed on the wrong side of a "cultural divide" in computing. But games proved to be elegant and useful metaphors for the very aspects of life that computing advocates wanted to influence.[1]

Prior knowledge

You do not need any prior knowledge of computer programming to learn from this book. In fact, if you have *no* prior knowledge of programming, then that is great! This book is designed for you!

You *do* need to have some basic knowledge of how to use your computer. Specifically, you should know how to do the following things:

- You should understand how to **right-click** (also known as **opposite-click**) on an object with your mouse.
- You should know how to **drag and drop** items.
- You should understand the basic locations in your file system and the basic operations on files.
  - You should know how to find the documents that you create, whether they are stored on your desktop, in a **Documents** folder, or elsewhere in your file system.
  - You should know where downloaded files are stored on your computer.
  - You should know how to copy files and how to move files on your computer.
- You should know the basics of browsing the Internet with your web browser.

- You should know the purpose of basic components of a computer system, including *hardware*, *software*, *RAM*, and *CPU*.

As long as you have those basic skills, you will get along just fine with this book!

Book web site and link codes

Throughout this book, I refer to *link codes* when referring to Internet resources. I have done this for simplicity. The codes are easier to type compared to lengthy URLs, and they can be updated in the event that a URL changes.

You can use the codes in a few ways. The easiest way is to use the book's website. Open the following URL in your favorite web browser:

`https://books.robotech.app`

Click on the book cover to get to the support page for this book. You can enter link codes on this page. Feel free to add this page to your web browser's bookmarks or favorites.

Another way to use the link codes is to add the link code to this book's support URL. For example, to use the link code **B**-7, you could enter this URL in your favorite browser:

`https://books.robotech.app/part-zero/B-7`

The last way to use the link codes is to look them up in the master list of codes for this book. You can see all of the book's codes and their URLs here:

`https://books.robotech.app/part-zero/list-all`

Notes

[1] Hennessey, Jonathan, and McGowan, Jack. (2017). *The comic book story of video games: The incredible history of the electronic gaming revolution*, pp. 29–32. New York: Ten Speed Press.

Hello, world! Introducing MakeCode Arcade

We are about to embark on a journey through the history of video games. On this journey, you also will learn how to write computer programs! My name is Robo, and I will be your guide.

In this first chapter, we will learn about the programming environment that we will use throughout this book. By the end of the chapter, you will create your very first masterpiece!

If you have not already done so, be sure to flip back a few pages and read the introduction. There is some important information there about link codes and prior knowledge that you will need to know before beginning.

Learning objectives

By reading this chapter and completing its activities, you will master the content when you can do the following:

- List the system requirements for MakeCode Arcade.
- View a project and its code in MakeCode Arcade.
- Connect to the MakeCode Arcade environment.
- Create a new project in MakeCode Arcade.
- Save and load a MakeCode Arcade project.
- Describe hardware that can be used with MakeCode Arcade.
- Describe the history of MakeCode Arcade.

About Microsoft MakeCode

Microsoft describes its MakeCode product as “a free, open source platform for creating engaging computer science learning experiences that support a progression path into real-world programming.”[2] Microsoft has created several MakeCode products, including editors for the BBC micro:bit, Lego Mindstorms, and Mojang's Minecraft. You can learn more about Microsoft MakeCode by using the link code **A-MAKECODE**.*

About MakeCode Arcade

In this book, we will use one of the MakeCode products: MakeCode Arcade. Microsoft describes MakeCode Arcade as "a code editor to build Retro Arcade games for the browser and handheld consoles."[3] MakeCode Arcade supports three programming languages: Blocks, JavaScript, and Python. In this book, we will use the Blocks language to learn about **program logic**. *Program logic*, or simply *logic*, is the thought process behind how a program works. We also will learn about **program design**, which is how a program is structured.

The Blocks language is an appropriate metaphor for both logic and design. We often think of building programs as taking pieces of knowledge that we already know—the blocks—and putting them together in different ways to create different programs.

System Requirements

To create programs in MakeCode Arcade, you will need the following:

- A computer with a web browser
  - MakeCode Arcade is a web-based application, so it can be used on tablets, laptops, and desktops. While MakeCode Arcade can run on smartphones, their limited screen size will make code editing difficult. *Playing* MakeCode Arcade games on smartphones, however, works really well!
  - MakeCode Arcade can be used with any modern browser.
  - MakeCode Arcade is frequently tested with Apple Safari, Google Chrome, Microsoft Edge, and Mozilla Firefox.
  - MakeCode Arcade is tested with current versions of browsers, so be sure to keep your browser updated.
- A keyboard and a mouse

*For a reminder on how to use link codes, head to this book's introduction.

    - While you can create programs in MakeCode Arcade with a touchscreen and the on-screen keyboard for your device, you will find a better experience when using a keyboard and a mouse.
- An Internet connection
    - Because MakeCode Arcade is a web-based application, you will need an Internet connection to load the interface into your web browser.
    - Once MakeCode Arcade is loaded into your web browser, though, an Internet connection is not required for its core functionality. Note that some functionality within MakeCode Arcade, like loading games from the gallery and interacting with GitHub, requires an Internet connection even after the environment has been loaded.
    - Offline editors (i.e. editors that do not require an Internet connection) for MakeCode are available. Use link code **M-OFFLINE** to access the latest information on using MakeCode Arcade offline.

You can use the following hardware to enhance your experience when playing MakeCode Arcade games:

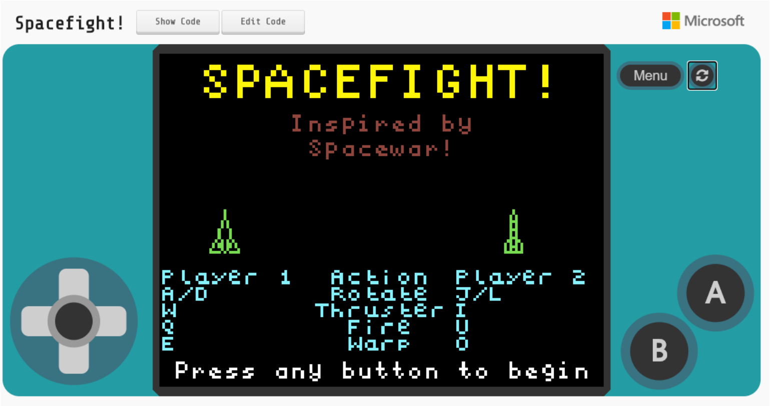
- Game controllers
    - MakeCode Arcade games can be written for up to four players. You can use the keyboard for up to two players, but for a better gameplay experience or to accommodate more than two players, you can use game controllers. Any game controller that is compatible with your computer should work in MakeCode Arcade: gamepads, joysticks, flight controllers, arcade controls, etc.
- Dedicated gameplay hardware
    - Besides playing games in your web browser, you can load your games onto separate hardware. Some of these devices are portable, so you can take your games anywhere! More information on these devices appears later in this chapter, in the “Working with hardware” section.

View a MakeCode Arcade project

We will begin our journey through the MakeCode Arcade interface by exploring a game. Use the following link code to launch a clone of *Spacewar!*

M-SW

A link like the one above is created in MakeCode Arcade to allow creators to share their games. When you open a link to a shared game, the game is loaded into your browser to allow you to play it. The interface looks like the screenshot in Figure A-B.



Player 1	Action	Player 2
A/D	Rotate	J/L
W	Thruster	I
Q	Fire	U
E	Warp	O

Playing Spacewar! via a shared link.

The main elements in the interface are described below. We will explore each of these a little later in this chapter.

- **Simulator**. As mentioned earlier in this chapter, MakeCode Arcade games can be played on dedicated hardware. MakeCode Arcade includes a simulator that acts like that dedicated hardware. In the simulator, you can see the game as it would be displayed on hardware, and you can interact with the game using the same buttons that are available on hardware.
- **Code viewer**. You can see the game's code by clicking on **Show Code**. Here, you can inspect the code and see how the game's author

was able to build the various elements of the game. You can learn a lot just by reading code!

- **Code editor**. You can load the game's code into your own editor, too! When you click on **Edit Code**, you can alter the game any way you like, from changing the graphics to fixing bugs to adding new gameplay.

Start your engines: Playing games in MakeCode Arcade

The game loads in the simulator so that you can play it. Maximize your browser or, better yet, switch to fullscreen mode in your browser by pressing **F11** on your keyboard. To exit fullscreen mode, press **F11** again.

This game has a **splash screen** that lists the game controls and the basic object of the game. In video games, a *splash screen* or *attract screen* runs while players are not interacting with the game. A splash screen can tell a story or provide information to the players before the game starts. For games that are displayed in an arcade, attract screens are used to draw players to the machine and play the game.

Lab A-B: Play *SpaceWar!*

Play *Spacewar!* for a little bit. Feel free to ask a friend to play against you! After you have played *Spacewar!* for a little while to learn the in-game controls, we will learn more about the MakeCode Arcade interface.

Game controls in MakeCode Arcade

There are several ways to interact with your computer to play games that are made in MakeCode Arcade. You can use game controllers connected to your system if you have them. You do not need game controllers, though, for one- or two-player games in MakeCode Arcade.

Player one can use the on-screen controls in the simulator to play a game. Player one also can use the keyboard. To use the on-screen controls, simply click (or tap if your system has a touchscreen) on the d-pad arrows or on the buttons to activate them. To use the keyboard, refer to Table A-B for the keys that you can use when playing a game.

Keyboard controls for player one.

Player 1	Option 1	Option 2
Up	W	Up arrow
Down	S	Down arrow
Left	A	Left arrow
Right	D	Right arrow
A	Q	Space
B	E	Enter

Player two can use the keys listed in Table A-B.

Keyboard controls for player two.

Player 2	Key
Up	I
Down	K
Left	J
Right	L
A	U
B	O

History of Spacewar!

Spacewar! running on the Computer History Museum's PDP-1.

"Spacewar!-PDP-1-20070512" by Joi Ito from Inbamura, Japan is licensed under CC BY 2.0.

We will look at many games in this book. So, we will use the history of video games as a guide for our exploration. This history will provide some context to the games that we explore.

Our first stop in our history lessons is *Spacewar!*, originally developed from 1961 to 1962 at the Massacusetts Institute of Technology (MIT) by Steve Russell, with contributions by Martin Graetz, Wayne Wiitanen, Bob Saunders, and Steve Piner. While other games like *Bertie the Brain* and *OXO* (tic-tac-toe games), simulations of draughts[a] and chess, and *Tennis for Two* were created earlier, *Spacewar!* is arguably the first video game that was widely distributed.

Originally written for the PDP-1 minicomputer by Digital Equipment Corporation (DEC), *Spacewar!* was ported[b] to many other platforms. After Russell and his friends created the original version of *Spacewar!* for the PDP-1, the game was recreated on many other mainframe computers and minicomputers of the time. *Spacewar!* quickly found its way to microcomputers when they became popular in the 1970s. *Spacewar!* continues to be ported today to platforms like the Microsoft Xbox 360 and, now, Microsoft MakeCode Arcade!

Spacewar! is so important in the history of video games that it was named among the top ten video games to be archived by the United States Library of Congress. In 2018, the Academy of Interactive

Arts and Sciences awarded the Pioneer Award to the surviving contributors to *Spacewar!*

Much more about *Spacewar!* is available on the Internet. You can read an article written by one of *Spacewar!*'s contributers, Martin Graetz, by using the link code **A-SW**.[4] You can play an emulated version of the original game by using the link code **G-SW**.

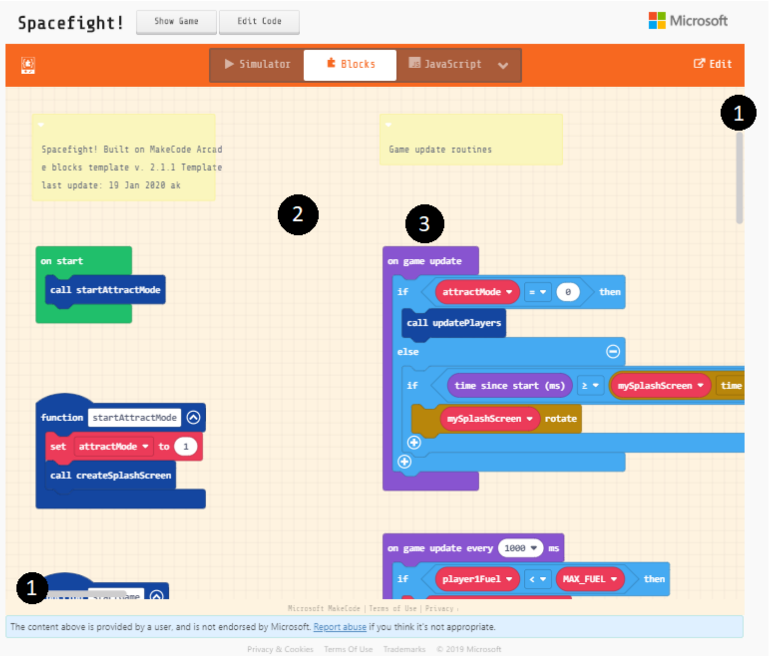
[a] In the United States, the game of *draughts* is known as *checkers*.

[b] *Porting* software means to replicate a piece of software so that it runs on a different computer. The computer could run a different operating system, or it could have different hardware that requires the software to be written differently.

Peek beneath the hood: A look at the code

After you have played *Spacewar!* for a little while to get accustomed to the in-game controls, let's take a look at the code for this game. A game's *code* are the instructions that your computer uses to run the game. A creator writes code to add images to a game, to respond to player actions, to change the score, to display the attract screen, and everything else that the game needs to do.

Click the **Show Code** button at the top of the screen. Your screen should look similar to Figure A-B.



Spacefight! Built on MakeCode Arcade blocks template v. 2.1.1 Template last update: 19 Jan 2020 ak

Game update routines

```
on start
    call startAttractMode

function startAttractMode
    set attractMode to 1
    call createSplashScreen

on game update
    if attractMode = 0 then
        call updatePlayers
    else
        if time since start (ms) ≥ mySplashScreen time
            mySplashScreen rotate

on game update every 1000 ms
    if player1Fuel < MAX_FUEL then
```

Microsoft MakeCode | Terms of Use | Privacy

The content above is provided by a user, and is not endorsed by Microsoft. Report abuse if you think it's not appropriate.

Privacy & Cookies Terms Of Use Trademarks © 2019 Microsoft

Viewing the code for Spacewar! in MakeCode Arcade.

There is a lot of code in this game! Do not become intimidated by the code that you see. In time, and with some practice, you will be able to create programs that are just as complex … and even more so!

Some of the things to investigate in the **Show Code** interface are described below:

1. **Scroll bars**. Use the scroll bars to move around the canvas.
2. **Canvas**. The *canvas* is where blocks are placed to write code for the game.
3. **Blocks**. *Blocks* are used to build the code for the game.

Lab A-B: Explore *Spacewar!* code

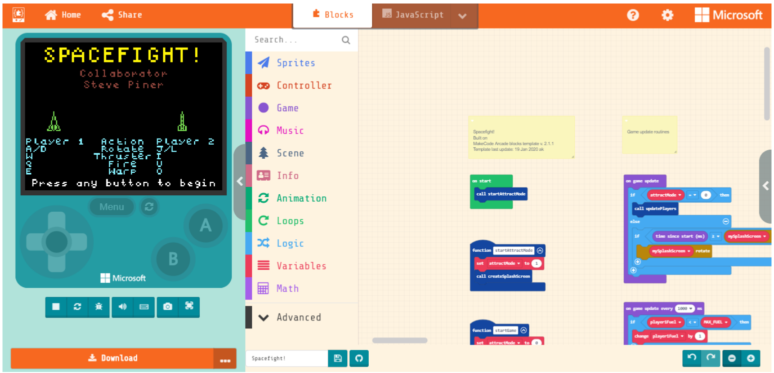
Take some time to explore the code blocks that comprise this game. Look at the different shapes and colors of blocks, how the blocks are organized and arranged, and the words in the different blocks. Just by looking at the different blocks and by reading them, see if you can make some assumptions on what the different blocks do and how they affect the game.

Feel free to go back and forth between playing the game in the **Simulator** interface and looking at the code in the **Show Code** interface. As you explore the code, take some screenshots and jot down some notes on different blocks. Perhaps you want to understand a particular block more. Perhaps you have an idea of what a certain block does and you want to remember it for later. Perhaps you want to test a hypothesis by changing some code and seeing its effects.

You probably will have many questions, and that is OK! By the time you reach the end of this book, you will be able to revisit *Spacewar!* and understand all of the code blocks that you see.

Time for a tune-up: Editing a game in MakeCode Arcade

If you would like to tinker with the code, click the **Edit Code** button at the top of the interface. This opens the code in MakeCode Arcade so that you can edit it. Your screen will look similar to the screenshot in Figure A-B.



```
Spacefight!
Built on
MakeCode Arcade blocks template v. 2.1.1
Template last update: 19 Jan 2020 ak

Game update routines

on start
    call startAttractMode

function startAttractMode
    set attractMode to 1
    call createSplashScreen

function startGame
    set attractMode to 0

on game update
    if attractMode = 0 then
        call updatePlayers
    else
        if time since start (ms) ≥ mySplashScreen
            mySplashScreen rotate

on game update every 1000 ms
    if player1Fuel < MAX_FUEL then
        change player1Fuel by 1
```

Spacewar! code open in the MakeCode Arcade editor.

This is now your own copy of the code, and you can change it however you like. Refer to the notes that you made from the previous section, and investigate the blocks that looked interesting to you. As you go through this book, we will get to know all of the controls that you see in the editor interface. For now, though, just explore on your own.

Consider following these steps when you change the code:

1. **Make note of the code's current state** before changing the code. Use a screenshot or jot down notes so that you can put the code back the way it was, particularly if you make an error.
2. **Make note of the changes that you want to make**. Jot down the changes that you plan to make to the code, why you want to change the code, and what you expect the changes to do.
3. **Make the changes**. Alter the code as you planned.
4. **Run the game and observe the new behavior**. Did the game change as you expected it to? How so, or how not?
5. **Return the code to its original state**. If you wish, return the code to its previous state. You do not have to, of course, particularly if you want to see how multiple changes affect the game. It's your code now! Have fun!

Lab A-B: Edit *Spacewar!* code

Spend as much time as you like in the editor. If you make an error and need to open the original code, simply close MakeCode Arcade, and then use the link code **M-SW** to open a new copy of the game.

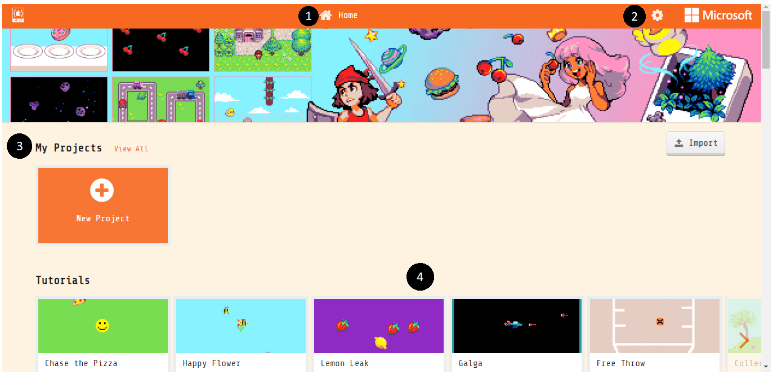
Start MakeCode Arcade

Now that you have played with someone else's code, we will open the MakeCode Arcade environment without code so that you can explore more of the interface.

You can open the MakeCode Arcade environment from the book's web site. You also can use the link code **M**, or you can visit the following URL:

https://arcade.makecode.com

When you open the MakeCode Arcade environment without opening a shared game, you will see something similar to the screenshot in Figure A-B.



The MakeCode Arcade home screen.

Some of the elements of the MakeCode Arcade home screen are described below:

1. **Home button**. Just about anywhere within the MakeCode Arcade environment, you can click on the **Home** button to return to this home screen.
2. **Settings cog**. Click on the settings cog to adjust the settings of MakeCode Arcade. We will explore different settings as we need them.
3. **My Projects**. As you create projects in MakeCode Arcade, they will appear in the **My projects** section. You may already have a project if you opened the *Spacewar!* game in the editor.
4. **Gallery**. Below the **My Projects** section is the gallery. There are a lot of built-in projects here! You will get a chance to explore the gallery in just a moment.

Built-in projects

Take some time to explore the gallery on the MakeCode Arcade home screen. For now, do not open any of the projects in the sections called **Tutorials**, **Multipart Tutorials**, or **Game Design Concepts**. We will work with tutorials in the next chapter.

There are many projects that you can explore. Some of the projects were created by the MakeCode Arcade team at Microsoft. Others were created by members of the MakeCode Arcade community. Click on any of the gallery images to get an explanation of the project and a link to the game. Some games will open directly in MakeCode Arcade. Others will take you to the MakeCode forums, which are described later in this chapter.

Lab A-B: Explore the MakeCode Arcade gallery

Feel free to open any of the games in the gallery (again, except for the tutorials; we will come back to those). As with *Spacewar!*, once you have opened a game in the editor, feel free to alter any of the code to see what the changes will do. As before, you may not understand all of the code that you see, and that is OK. Some of the games in the gallery are very complex! Take your time and enjoy the exploration.

Your first MakeCode Arcade project

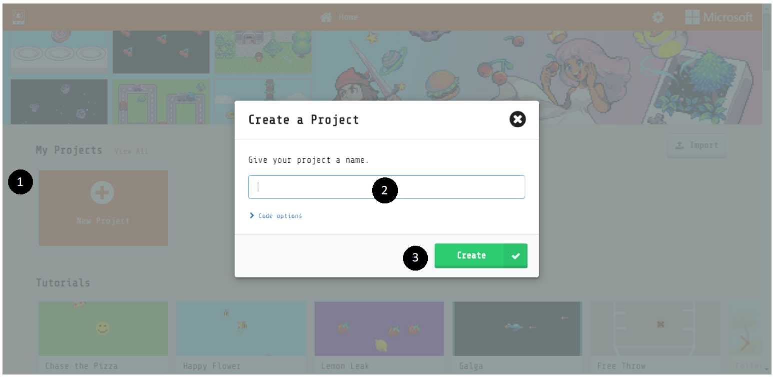
Now that you have spent some time looking at code from other creators, it is time to create your own project. You will create a simple game with two images. The player will be able to move one of the images with the player controls.

Return to the MakeCode Arcade home screen by clicking on the **Home** button. If you no longer have the MakeCode Arcade editor open, simply visit the MakeCode Arcade home screen by any of the methods listed earlier in this chapter.

Creating a new project in MakeCode Arcade

Lab A-B: Create a new project

From the MakeCode Arcade home screen, create a new project.

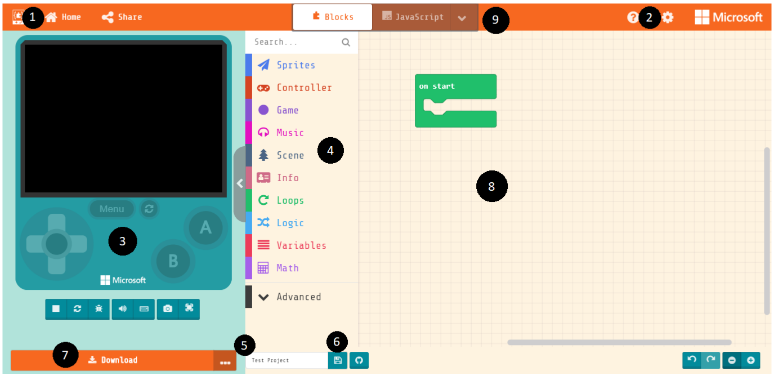


Creating a new project in MakeCode Arcade.

1. In the **My Projects** section of the home screen, click on the **New Project** button.
2. When prompted, give your project a name.
3. Click on the **Create** button.
4. The MakeCode Arcade editor opens with a blank canvas.

Exploring the MakeCode Arcade editor

Figure A-B is a screenshot of the MakeCode Arcade editor with a blank canvas (i.e. a blank or empty project). In this section, we will explore some of the components of the interface in more detail.



MakeCode Arcade code editor with a blank canvas.

1. **Home button**. The **Home** button will return you to MakeCode Arcade's home screen.
2. **Settings cog**. Use the **settings cog** to adjust different settings in MakeCode Arcade. You also can use the settings cog to print your project and to delete the current project.
3. **Simulator**. The **simulator** plays the current game. You learned how to play games in the simulator earlier in this chapter. When editing a project, there are additional controls beneath the simulator. We will learn to use those controls in later chapters.
4. **Toolbox drawers**. There are many blocks in the MakeCode Arcade environment. The blocks are stored in different drawers in the toolbox. Click on a drawer to open the toolbox and see the blocks in that drawer.
5. **Project name**. You can change the name of your project with this text box.
6. **Save button**. You can save a copy of your code as a cartridge with the **save** button. You will learn more about cartridges later in this chapter.
7. **Download button**. MakeCode Arcade can send your code to a hardware device with the **Download** button. You will learn more about hardware for MakeCode Arcade later in this chapter. Follow your hardware's instructions on how to download code to the device.
8. **Canvas**. The **canvas** contains the code for your project. You add blocks to the canvas, and you can drag blocks around the canvas to organize them. The canvas grows automatically as you need more room for your code.
9. **Language buttons**. You can use the **language buttons** to show your code in different programming languages. We will use the Blocks language in this book. You can learn about other programming languages in other books in this series.

Planning the first project

Staring at a blank canvas can be intimidating. Where do we begin? For all projects, the best thing to do is to create a plan before writing any code. We will do that now for our first project.

For many projects, we can create a plan by considering the following parts:

- **Project name**. A descriptive name for the project.
- **Project description**. The project description provides an overview of the project.
- **Housekeeping**. Housekeeping items are things that we need to create before the game begins, like the player's image, a timer, and the game score.
- **Events**. Events are things that happen in your game as a result of other actions, such as player input or when a player earns a certain score.

We will explore these parts of the project plan in more detail in other projects. We also will expand this project plan as our projects get more complex. In future projects, we will create the plan together. However, for this first project, I will provide you with the plan that we will follow.

Project Plan

Project name: Control a sprite

Project description: For this project, we are creating a game with two images. One of the images will be controlled by the player, and the image will move around the screen based on the player's actions.

Housekeeping: Two sprites.

Events: One sprite will move based on the player's actions.

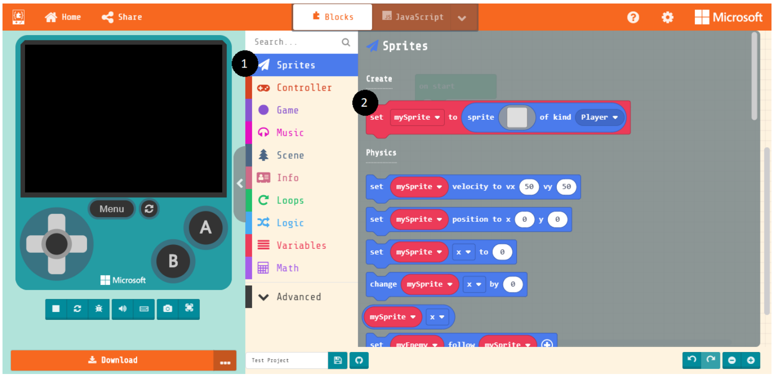
Adding the first block

In this book, we will be using blocks to create our programs. After finishing this book, you can learn how to use other languages in MakeCode Arcade.

Let's add a block to the project. We know we need to add an image that the player can control. For this, we will add a **sprite**. A *sprite* is an image with some additional properties that can be manipulated during the game. For now, you will learn just a little bit about how sprites work. You will learn more about sprites in the next chapter.

Lab A-B: Add your first block to your new project

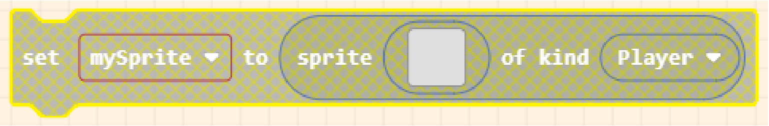
Follow these steps to add your first sprite to your project:



Adding the first sprite to your blank project.

1. Click on the **Sprites** drawer. The sprites toolbox opens.
2. Click on the **set sprite** block in the toolbox. A copy of the block appears on your canvas.

Great! We have added our first block to our project. Something looks wrong, though, doesn't it? The new block seems to be crossed out:



A disabled block.

Block types

MakeCode Arcade uses several types of blocks. The **set sprite** block that you just placed on the canvas is called a **single block**. Single blocks cannot live independently on the canvas. Instead, single blocks must live inside of another type of block, called **container blocks**. Certain container blocks, called **event blocks**, can be placed anywhere on the canvas. You will learn more about these types of blocks as you progress through the book.

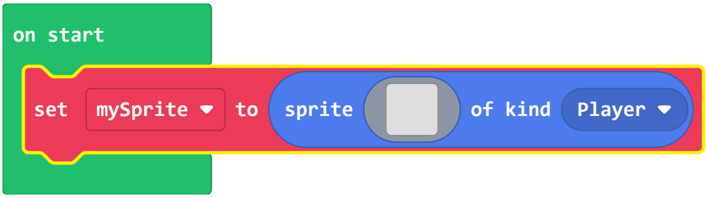
The *on start* block

For now, though, we are going to use an event block that is used in most games: the **on start** block. The **on start** block is the first block that runs at the beginning of your program. MakeCode Arcade games can only contain one **on start** block, and this often is where we place housekeeping items before gameplay begins.

According to our project plan for this first project, we will need to add a couple of sprites to our game, and we need to configure one of the sprites to respond to the player's inputs. We will add these items to the **on start** block of our program.

Lab A-B: Adding first block to *on start*

The **on start** block is automatically added to your project when MakeCode Arcade creates it. Simply drag your **set sprite** block into the **on start** block so that it runs at the beginning of your game. When you drag the block into the **on start** block, notice that it is not longer disabled.



```
on start
  set mySprite ▾ to sprite □ of kind Player ▾
```

*Adding the **set sprite** block to **on start**.*

Look at the shape of the **on start** block. Notice the differences between it and the **set sprite** block? Because of its shape, you can tell that the **on start** block is a container block, and you can place other blocks inside of it. Also, the **on start** block is an event block. You cannot tell that by its shape; you simply learn which blocks are event blocks and which ones are not. In case you forget, there is an easy way to tell which blocks are event blocks. You already have used this technique. Simply drop the block onto the canvas. If the block is disabled when you place it by itself on the canvas, then it is not an event block.

Renaming a variable

Read the **set sprite** block. It contains the following code:

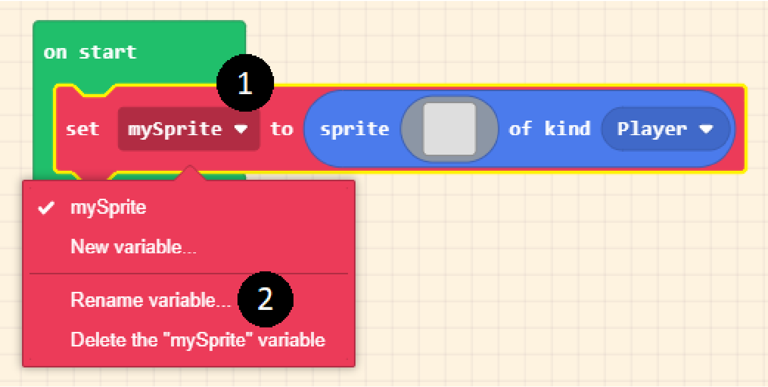
```
set mysprite to sprite ' ' of kind Player
```

The default name for your sprite is *mysprite*. That does not sound very descriptive, does it? If you place another **set sprite** block in your program, can you guess what its name will be?

By adding the **set sprite** block to your program, you have created a **variable**. A *variable* is a place in the computer's memory which is given a name. We can store information in variables; in this case, we are storing a sprite. When we write programs, we often create many variables. To help us keep better track of our variables, we assign them meaningful names. The name *mysprite* is not very meaningful, so let's rename our variable to give it more meaning.

Lab A-B: Rename the first sprite

This first sprite will be the one that is controlled by the player. With that in mind, what name do you want to give it?


on start
set mySprite to sprite of kind Player
1
✓ mySprite
New variable...
Rename variable...
2
Delete the "mySprite" variable

Renaming a variable.

1. Click the down arrow next to the variable name, *mysprite*. A menu appears.
2. Look past the list of variable names that already exist in your project. Near the bottom of the menu, click **Rename variable...**. A dialog box appears.
3. Enter your new variable name, and then click **OK**.

Autosave in MakeCode

Now is a good time to take a break from our first project. You have learned a lot already!

MakeCode automatically saves your work. We can see that now: Click on the **Home** icon to return to the home screen, and you can see your new project in the **My Projects** space.

On the home screen, click on your new project, and you are returned to the editor. This time, though, instead of a blank canvas, you are returned right where you left off with your new project. No manual saving necessary!

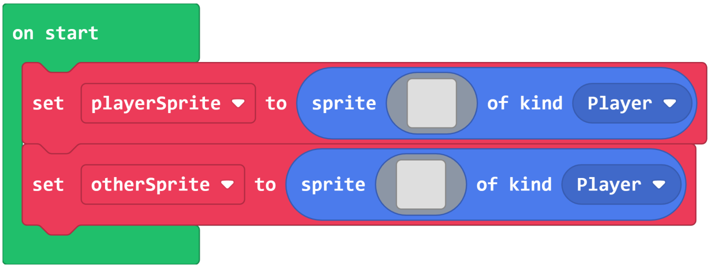
MakeCode stores your projects in your browser's **data store**. The data store a location on your hard drive where web applications can keep data. Because this information is kept in your browser, you will lose your projects (and any other data that MakeCode stores) if you clear your browser's data store. Also, if you use a different browser or a different computer, you will not see your projects.

Later in this chapter, in the section called "Working with projects," we will learn another way to store your projects.

Adding another sprite

Lab A-B: Add the second sprite to your project

Open your project again in the MakeCode Arcade editor. Let's add a second sprite.



```
on start
    set playerSprite ▾ to sprite □ of kind Player ▾
    set otherSprite ▾ to sprite □ of kind Player ▾
```

Adding a second sprite to your project.

1. Click on the **Sprites** drawer. The sprites toolbox opens.

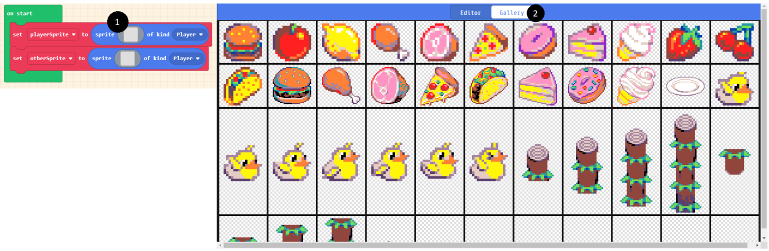
2. Click on the **set sprite** block in the toolbox. As before, a copy of the block appears on your canvas.
3. Drag the block into the **on start** block.
4. Rename the variable to give it a meaningful name.

Using images from the gallery

Take a look at our game in the simulator. If the simulator is not already running, then click on the **Play** button. We can see from the code that we have added two sprites to our game, but in the simulator, it looks like nothing is happening.

When we created our two sprites, we did not change the **default** image. A *default* is a value that is set automatically. For sprites, the default image is blank or empty. Let's change that now by using the image gallery.

Lab A-B: Add images from the image gallery



```
on start
  set playerSprite ▾ to sprite [ ] of kind Player ▾
  set otherSprite ▾ to sprite [ ] of kind Player ▾
```

Using the image gallery for sprites.

1. Click on the image box for the player's sprite. The image editor opens.
2. Click on **Gallery**.
3. Select an image. It is loaded into the image editor
4. Click on **Done** in the bottom right corner.

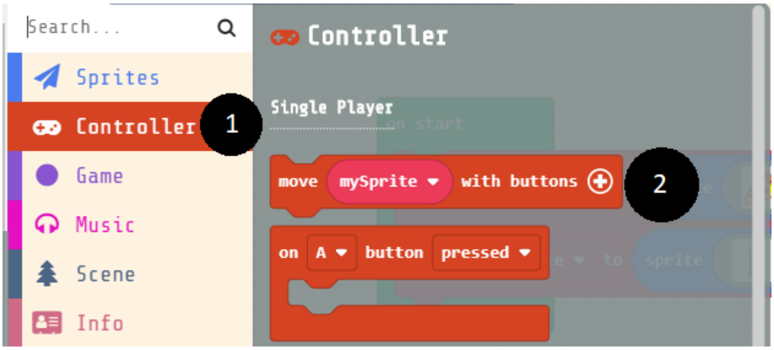
After you set the image for the player's sprite, the simulator restarts, and your new sprite should appear in the simulator.

Do the same for the second sprite. After you set the image for the second sprite, the simulator will restart again. Now, you should see both sprites in the simulator, and they should overlap.

Attaching controls to a sprite

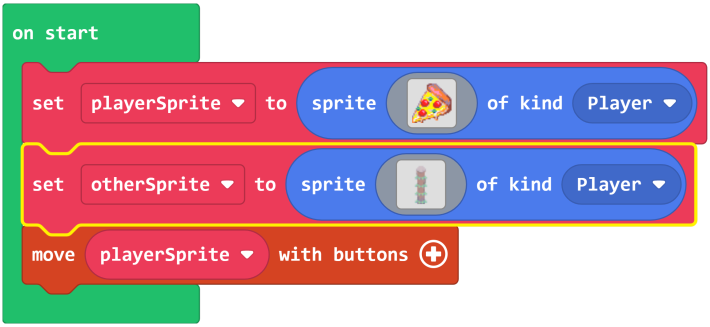
To finish our first project, we need to add the ability for the player to control one of the sprites. MakeCode Arcade makes that easy! Follow these steps:

<u>**Lab A-B: Add controls to the player sprite**</u>



```
move mySprite ▾ with buttons ⊕
on A ▾ button pressed ▾
```

Adding player controls to a sprite.



```
on start
    set playerSprite ▾ to sprite [ ] of kind Player ▾
    set otherSprite ▾ to sprite [ ] of kind Player ▾
    move playerSprite ▾ with buttons ⊕
```

Final code for our first project.

1. Click on the **Controller** drawer.
2. Click on the **move sprite with buttons** block in the toolbox.
3. Drag the new block to the bottom on the **on start** block.
4. In the **move sprite with buttons** block, set the sprite variable to the one that you want the player to control.

Allow the simulator to restart (press the play button if the simulator is paused). Now, you can control one of the sprites with the arrow keys!

Great work! You have created your first project!

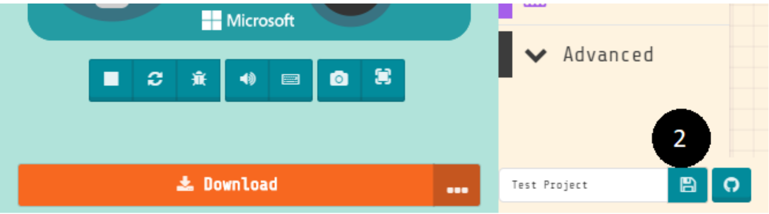
Working with projects

Saving your project as a cartridge

As mentioned earlier, MakeCode Arcade saves your work automatically to your browser's data store. This is a useful feature of MakeCode Arcade, but it can fail. Because MakeCode depends on your browser's data store, all of your work can quickly be erased by clearing your browser's data store. So, do not think of this as a permanent storage space for your projects. You should get in the habit of saving your work elsewhere.

You can save a copy of your project anytime it is open in the editor. MakeCode Arcade can save your project as a **cartridge**. In MakeCode Arcade, a cartridge is a special image file with your code stored inside of it. MakeCode Arcade saves cartridges as PNG files.

Lab A-B: Create a cartridge



Creating a cartridge in MakeCode Arcade.

Follow these steps to create a cartridge of your first project:

1. Open your project in the editor.
2. Near the bottom of the screen where your project name is displayed, click on the **Save** icon.
3. Your browser will download the cartridge and store it in your Downloads folder.
4. Move the cartridge file to an appropriate location in your file system.

Keep in mind that this cartridge is a copy of your current code. If you go back into the editor and change any of your code for this project, then your changes will not be reflected in the cartridge. This is a good way to keep previous versions of your projects. Before you make any major changes to a project, save a copy of it. If you make a major error, you can always load the old copy of your code.

Loading a cartridge

Lab A-B: Load a cartridge into MakeCode Arcade

Let's load this cartridge that you just created into MakeCode Arcade.

1. Return to MakeCode Arcade's home screen by clicking on the **Home** button.
2. Drag your cartridge file into your browser.

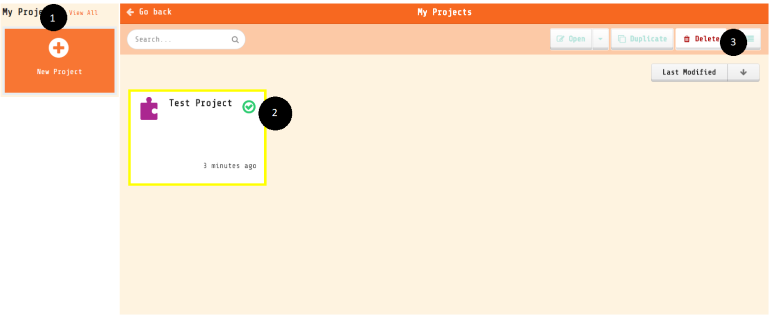
1) First, MakeCode Arcade creates a new project in your browser's data store.
1) Then, it copies the code in your cartridge into the new project.
1) Then, it opens the new project in the editor.
1. Play your new copy of your project for a few moments to make sure it works correctly.

Return to the home screen. Notice that you now have two copies of your project in the **My Projects** area. We do not need this extra copy, so let's delete one of them.

Deleting a project

Deleting a project removes it from your browser's data store. Remember that, for this first project, we already have created a cartridge of it. We can safely delete the project from our browser's data store because we have a copy of it saved on our hard drive.

Lab A-B: Delete a project from MakeCode Arcade



Deleting a project from MakeCode Arcade.

Follow these steps to delete a project from MakeCode Arcade:

1. From the home screen, click **View All** next to the **My Projects** label.
2. Click on the project that you wish to delete. Notice the green check mark to indicate that the project is selected.
3. Click **Delete**.
4. Confirm that you wish to delete the project by clicking **Delete** in the dialog box.

Notice that MakeCode Arcade warns you before it deletes a project. Deleting a project from your browser's data store is a permanent action; there is no undo. Be sure you save a project as a cartridge before you delete it from MakeCode Arcade.

Working with hardware

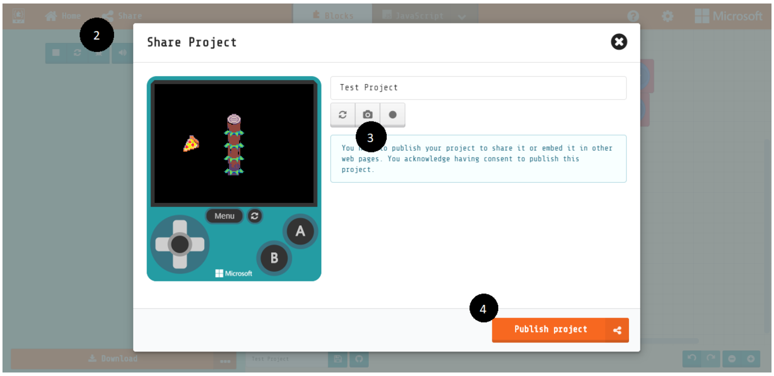
To work with projects in MakeCode Arcade, all you need is a system with a modern web browser. MakeCode Arcade includes a simulator that will run your projects, and you can play games that others create in the simulator.

However, you also can play games on separate hardware that is specifically designed or configured for MakeCode Arcade. Several manufacturers have created boards that work with MakeCode Arcade, including the BrainPad Arcade and the Adafruit PyGamer. Other general-purpose hardware, like the Raspberry Pi and the Adafruit M4, can be configured to run MakeCode Arcade games. You can even build a full-sized, upright arcade cabinet for your MakeCode Arcade games! For more information on hardware that supports MakeCode Arcade, use the link code **M-HW**.

Sharing your projects

There are several ways to share your projects with others. You already have learned to create a cartridge for your game. You can share that file with others, and they can load your cartridge into MakeCode Arcade. Another way to share a project is to send a copy to the MakeCode servers to get a custom URL.

Lab A-B: Share your new project



Test Project

You need to publish your project to share it or embed it in other web pages. You acknowledge having consent to publish this project.

Generating a URL for your project.

Follow these steps to send a copy of your project to the MakeCode servers:

1. Open your project in the MakeCode Arcade editor.
2. Click the **Share** button. A dialog box opens.
3. Play your game in the simulator. When you find a good screenshot for your game, click on the camera icon.
4. Click the **Publish project** button.
5. Copy the URL for your game and paste it somewhere for future use.

Now that you have a URL for your project, you can share it however you like. You could post the URL to your blog. You can e-mail the link to friends. You can share the link over social media. You can include the link in a digital portfolio. You also can share the link on the MakeCode forums.

MakeCode forums

The MakeCode team has created a Discourse forum for the MakeCode community. Use the link code **M-FORUM** to visit the MakeCode forum, or visit the following URL:

https://forum.makecode.com

In the MakeCode forums, you will find information on the live streams that the MakeCode Arcade team conducts. You also will find games and other projects that creators have shared. Many members of the MakeCode Arcade team participate in the forum, answering questions and sharing updates. You will often find me participating at the forum, too!

The forum is available to the public, but you will need to create an account in order to participate. If you wish to participate in the forum, it is probably best to create a GitHub account. GitHub is a widely-used service to host code, and you can use GitHub with MakeCode Arcade. We will discuss using GitHub in the *Part zero mark II* book.

Feel free to share your creations with the world through the MakeCode Arcade forum. Be sure to say “Hi!”

History of MakeCode Arcade

MakeCode Arcade’s immediate predecessor was TouchDevelop. TouchDevelop allowed creators to build programs using a Windows Phone. It later expanded to a web interface and allowed for input with a keyboard and mouse. Like MakeCode’s Blocks language, TouchDevelop used interlocking blocks to create programs.

TouchDevelop was retired in June 2019. Several of the team members behind TouchDevelop are also members of the MakeCode team.

Hello, it’s me! (Or is it I?)—Sprites and your hero character

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Learning objectives

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Additional controls in the MakeCode editor

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Deleting blocks

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Undo and redo

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Zooming in and out

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Working with tutorials

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Moving around

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

The hero character and sprite

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Player sprite velocity

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Constraining sprite movement

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Using the image editor

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Screen coordinates

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Exploring the screen coordinates

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Sprite coordinates

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Placing your character

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Animating your character

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Installing extensions

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Creating a walking animation

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Creating a spinning wheel

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

More complex animations

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Making a background

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Background with a solid color

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Background image

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Animated backgrounds

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Artwork and prototypes

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Additional practice with the image editor

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Answers to review questions

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Which way did it go? Decisions, conditionals, and basic game physics

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Learning objectives

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Software development life cycle

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Step one: Gathering requirements

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Step two: Project analysis

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Step three: Project design

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Step four: Coding

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Step five: Testing

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Step six: Operations

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Step seven: Project archiving

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Other software development life cycles

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Planning a project

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Project documentation

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Algorithms and structured programming

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Flowcharts

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Nassi-Shneiderman diagrams

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Pseudocode

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Princess Pizza

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Documentation in Blocks programs

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Bouncing Burger

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Event-driven programming

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Sprite types

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Pong

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Pong simplified

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Velocity

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Adding sound effects

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Determining the game’s end

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

The `on game update` block

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Boolean expressions

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Correcting an error

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Acceleration

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Decision structures

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Additional tutorials

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Answers to exercises

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

What’s the score? Variables and data types

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Learning objectives

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Which button?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Pong part 2: Complete one-player game

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Pong part 3: Complete two-player game

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Introduction to variables

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Creating variables

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Naming conventions

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Constants and literals

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Variable scope

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Global scope

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Local scope

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Data types in Blocks

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Number

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

String

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Sprite and image

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Boolean

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Guess My Number

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Acey Deucey

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Introduction to the Playing Cards extension

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Answers to exercises

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Zap! Debugging part 1

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

I’ve made some enemies—Introduction to object-oriented programming

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Visual effects

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Sprite effects

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Scene effects

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Debugging part 2

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Over … and over … and over again—Loops

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Who’s above me? Who’s next to me? Arrays and lists

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Better than right-click-duplicate—Functions

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Naming convention

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Local variables revisited

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Fun name :-) - Strings

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Writing text on the screen

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Fun name :-) - Tile maps and platformers

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Tile maps

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Platformers

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Mazes?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

What’s your name again? Persisting data

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Session object

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Stranger, danger! Security and privacy

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

It’s alive! Artificial intelligence

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Fun name :-) - Searching and sorting

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Where to now?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Additional books in the series

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Learn other languages and programming environments

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Other games to try to build in MakeCode Arcade

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Other programming books

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

More on the history of video games

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Appendix - A Blocks style guide

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Project description

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Variable and function naming

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Function description

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Appendix - Common game structures

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Shoe of cards

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Dice

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Appendix - Stopwatch in Blocks

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Part I: Basic stopwatch

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Finished code

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Developing the project plan

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Building the project

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Next steps

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Part II: Advanced stopwatch

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Finished code

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Adding to the project plan

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Incorporate changes into code

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Next steps

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Appendix - Common game mechanics

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Shuffling

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Appendix - Advanced graphics

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Parallax (sprites with depth)

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Appendix - Multiplayer games

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Appendix - Vector math

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Revisiting velocity

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Velocity vectors

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Angles

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Measuring angles

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Review of right triangles

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Trigonomotric functions and right triangles

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Practice with velocity vectors

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Representing π

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Sprite travelling in a random direction

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Another challenge

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Final challenge

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Appendix - A template for Blocks games

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Splash screen

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Titles

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Headlines

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Instruction lists

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Adding images to the splash screen

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Attract mode

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Button handlers

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Appendix - Additional Blocks games (showcase)

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Additional clones

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Notes

[2] Microsoft. What is Microsoft MakeCode? https://www.microsoft.com/en-us/makecode/about

[3] Microsoft. About MakeCode Arcade. https://arcade.makecode.com/about

[4] Graetz, Martin (August 1981). "The origin of Spacewar." Creative Computing. Vol. 6 no. 8. pp. 56–67. ISSN 0097-8140.

Credits and licenses

Screenshots of Microsoft MakeCode Arcade include content owned by Microsoft Corporation. Used with permission from Mirosoft.

"Signed Pong Cabinet" by Chris Rand is licensed under CC BY-SA 3.0.

"Spacewar!-PDP-1-20070512" by Joi Ito from Inbamura, Japan is licensed under CC BY 2.0.

"Tennis for Two in 1959" by Brookhaven National Laboratory is released to the public domain.

"Magnavox Odyssey Console Set" by Evan-Amos is released to the public domain.

"TeleGames-Atari-Pong" by Evan-Amos is licensed under CC BY-SA 3.0.

Part zero mark II: Time to type!

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

About the cover

Brief contents

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Introduction to *Part zero mark II*

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

About the series

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Intended audience

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Titles in the series

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

About the book

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Prior knowledge

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Book web site and link codes

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Hello, World! I'm typing! Introducing JavaScript

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Transitioning from Blocks to JavaScript

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Toolbox

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

IntelliSense

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Tooltips

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Style guide

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

To grandmother's house—Revisiting topics from a JavaScript perspective

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

SDLC and documentation

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Sprites

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Variables and scope

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Functions

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Return of the Jedi—Object-oriented programming basics

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

‘Round and ‘round we go—Loops and arrays

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Hold this, will ya? Working with GitHub

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Introduction to agile methods

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Publishing with GitHub

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Our first game

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Our second game

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Our third game

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Fun title :-) - Advanced graphics and physics

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Parallax (Revisit of *Part zero* appendix)

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Drawing directly to the screen

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

HUD

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Simulating three dimensions

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Fun title :-) - Searching and sorting

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Fun title :-) - Introduction to data structures and algorithms

Where to now?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Additional books in the series

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Learn other languages and programming environments

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Other games to try to build in MakeCode Arcade

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Other programming books

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

More on the history of video games

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Appendix - A TypeScript style guide

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Appendix - Common game structures

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Shoe of cards

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Dice

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Appendix - Stopwatch in TypeScript

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Introduction

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

TypeScript

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Finished code

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Working with the Text Sprite in TypeScript

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Object-oriented design

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Creating the Stopwatch class

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Adding functionality to the Stopwatch class

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Extras

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Appendix - Common game mechanics

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Shuffling

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Appendix - Lambda and anonymous functions

Appendix - Filtering and iterating with method calls

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Appendix - A template for JavaScript Games

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Appendix - Additional JavaScript games (showcase)

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Credits and licenses

Bonus chapter: Can I be a computer scientist?

A question that I get asked frequently is, “What do I need to know to be a computer programmer?” Or, more simply, “Can I become a computer scientist?” Here are my answers to these and related questions, based on my own experience as a computer scientist as well as working with others in this profession.

Can I be a computer scientist?

Of course! Just like anything else, some people will find it a little easier than other people. However, if you find it interesting, and you are willing to work hard at your skill, you *absolutely* can be a computer programmer.

If you find computer programming interesting, please pursue computer science as a profession. The world is in desperate need of high-quality programmers.

Do I need to be good at math?

Yes … sorta. When I get asked this question, it usually translates to “Do I need to do well in math classes?” You can be good at math and be *terrible* in math classes … and that usually is because you have had teachers that do not share their knowledge in a way that translates well to you. Nearly everyone can learn math, and nearly everyone has an intrinsic understanding of the basics of mathematics. We all learned how to count, and we all learned how to add, at *very* young ages, often before entering school. Math is built into almost all of us. Some of us are just able to tap into that basic understanding a little bit better than others.

To be a good computer scientist, you *do* need to be good at math … in that you need to be good at some fundamental skills in math:

- You need to be good at solving problems. (No, I don’t mean “word problems” or “story problems.”)
- You need to be able to recognize patterns.

You do *not* need to be an “A” student in your math classes, though.

I really like computers, but I don’t really like programming. Are there other things that I can do?

Absolutely! Computing is a large profession; not everyone involved in the profession is a computer scientist. The profession needs people to do many different things!

If you like computers, but programming doesn’t seem like your cup of tea, then perhaps *information technology* is more your speed. IT includes areas like network administration and systems administration (i.e. working with “servers”). Or, if you are more artistic-minded, then you might be interested in computer-aided design (e.g. architecture, 3-D modeling, graphics design). If you like building things, then automation (e.g. robotics) may be of interest. If you like numbers, then you may be interested in becoming a data scientist.

Most importantly, find something that is interesting to you. The more interesting something is to you, the more you will be willing to work hard at it, and the better you will get at your craft.

Glossary

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Timeline of video games

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Acknowledgements

Thanks to MakeCode Arcade forum user `rymc88` for the inspiration behind some of the content in the *Part zero* appendix on vector math.

About the author

Alex Kulcsar (“Robo” in his college days) is chair of the computer science programs at American Business & Technology University. He is also CEO and CTO of Robo Technical Group. He holds a bachelor’s degree in mathematics from the University of Michigan and a master’s degree in computer science from American Sentinel University.

Colophon

The preview versions of this book are created in Leanpub with Visual Studio Code. The final print and electronic versions are produced by IngramSpark.

Screenshots were taken in Microsoft Edge.

Link codes

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/makecode-book.

Bibliography